COMP 301 Analysis of Algorithms
Instructor: Zafer Aydın
Lab Assignment 7

Introduction

In this lab you will compare the running times of three matrix multiplication algorithms. Submit your answers to the questions below in a text file (e.g. Word document). Name your file in name_surname.docx format. Submit your solution document and Java codes as a compressed folder (.zip, .rar) in name_surname format to Canvas.

You can use the code templates in `matrix.java` in this lab.

Problem Statement

Given two matrices A and B each with size $n \times n$ compute $C = A.B$, which is the product of A and B.

Assignment

1. Choose different matrix sizes (n is a power of 2) starting from $n = 2$ up to $n = 128$. Fill the table below and report the running times of `square_matrix_multiply (SQMM)`, `square_matrix_multiply_recursive (SQMMR)` and `strassen_matrix_multiply (STMM)` methods in nanoseconds. Does Strassen's algorithm perform better than the other algorithms for the given matrix sizes considering the fact that the asymptotic running time of Strassen's algorithm is better? If Strassen's algorithm is slower than the other algorithms, state the reason for this behavior.

n	SQMM	SQMMR	STMM
2			
4			
8			
16			
32			
64			
128			

COMP 301 Analysis of Algorithms
Instructor: Zafer Aydın
Lab Assignment 7

2. (a) What modification can you make to Strassen's algorithm if the input matrix sizes are not powers of 2? Implement this modification as a separate method. (Hint: you can complete the odd number of rows to even by padding rows or columns of zeros to submatrices during recursion). Below is the original Strassen's algorithm

1. Divide the input matrices A and B and output matrix C into $n/2 \times n/2$ submatrices, as in equation (4.9). This step takes $\Theta(1)$ time by index calculation, just as in SQUARE-MATRIX-MULTIPLY-RECURSIVE.
2. Create 10 matrices $S_1, S_2, \ldots, S_{10}$, each of which is $n/2 \times n/2$ and is the sum or difference of two matrices created in step 1. We can create all 10 matrices in $\Theta(n^2)$ time.
3. Using the submatrices created in step 1 and the 10 matrices created in step 2, recursively compute seven matrix products $P_1, P_2, \ldots, P_7$. Each matrix P_i is $n/2 \times n/2$.
4. Compute the desired submatrices $C_{11}, C_{12}, C_{21}, C_{22}$ of the result matrix C by adding and subtracting various combinations of the P_i matrices. We can compute all four submatrices in $\Theta(n^2)$ time.

$$A = \begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix}, \quad B = \begin{pmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{pmatrix}, \quad C = \begin{pmatrix} C_{11} & C_{12} \\ C_{21} & C_{22} \end{pmatrix}$$

$$\begin{aligned}
S_1 &= B_{12} - B_{22}, \\
S_2 &= A_{11} + A_{12}, \\
S_3 &= A_{21} + A_{22}, \\
S_4 &= B_{21} - B_{11}, \\
S_5 &= A_{11} + A_{22}, \\
S_6 &= B_{11} + B_{22}, \\
S_7 &= A_{12} - A_{22}, \\
S_8 &= B_{21} + B_{22}, \\
S_9 &= A_{11} - A_{21}, \\
S_{10} &= B_{11} + B_{12}.
\end{aligned}$$

$$\begin{aligned}
P_1 &= A_{11} \cdot S_1 \\
P_2 &= S_2 \cdot B_{22} \\
P_3 &= S_3 \cdot B_{11} \\
P_4 &= A_{22} \cdot S_4 \\
P_5 &= S_5 \cdot S_6 \\
P_6 &= S_7 \cdot S_8 \\
P_7 &= S_9 \cdot S_{10}
\end{aligned}$$

$$C_{11} = P_5 + P_4 - P_2 + P_6.$$

$$C_{12} = P_1 + P_2$$

$$C_{21} = P_3 + P_4$$

$$C_{22} = P_5 + P_1 - P_3 - P_7$$

(b) Verify that your method works correctly for $A = \begin{bmatrix} 1 & 1 & 1 \\ 2 & 2 & 2 \\ 3 & 3 & 3 \end{bmatrix}$, $B = \begin{bmatrix} 1 & 1 & 1 \\ 2 & 2 & 2 \\ 3 & 3 & 3 \end{bmatrix}$, where C should be $= \begin{bmatrix} 6 & 6 & 6 \\ 12 & 12 & 12 \\ 18 & 18 & 18 \end{bmatrix}$.